IN THE SPECIFICATION:

Please amend the specification as follows.

Please amend the paragraph on page 3, lines 3-4:

FIG. 3 is a flow chart describing operations performed by the parallel asynchronous debugger in connection with the invention.

FIG. 3A is a flow chart of operations performed in handling a thread deletion request in accordance with an embodiment of the present invention.

FIG. 3B is a flow chart of operations performed in handling a break point instruction in accordance with an embodiment of the present invention.

Please amend the paragraph beginning on page 5, line 24, through page 6, line 7 as follows:

With this background, operations performed by the debuuger will be described in connection with the flow chart depicted in FIG. 3. Generally, the execution environment 20, operating system 23 and debugger 24 are started and initialized in a conventional manner (step 100). Thereafter, the operating system 23 and debugger 24 can control initialization of the program 21, with the debugger starting at least one thread 22(t) (step 101). If the program 21 issues a thread creation request requesting creation of a thread (step 110), the thread creation request is passed to the debugger 24 (step 111), which can create the thread (step 112) and enable it to start execution (step 113). Turning now to FIG. 3A, a flowchart of operations performed in handling a thread deletion request in accordance with an embodiment of the present invention is presented. ~~Similarly , if~~ If the program 21 issues a thread deletion request requesting deletion of a thread (step 120), the thread deletion request is passed to the debugger 24 (step 121), which can delete the thread (step 122).

Please amend the paragraph on page 6, lines 8-15 as follows:

Referring now to FIG. 3B, a flowchart of operations performed in handling a break point instruction in accordance with an embodiment of the present invention is presented. ~~On the other hand, if~~ If a thread $22(t_B)$ executes a breakpoint instruction (step 130), which may result in a trap to the operating system 23, the operating system 23 will typically stop operation of all of the threads $22(_{t})$ (step 131) and transfer control to the debugger 24 (step 132). In that case, the debugger 24 assumes control (step 133) and identifies the thread $22(t_B)$, which contained the breakpoint instruction (step 134). After the debugger has identified the thread $22(t_B)$, it enables the other threads $22(t_1)$, $22(t_2)$, … $(t_1, t_x, \ldots \neq t_B)$ to resume operations (step 135), and allows the operator to control subsequent processing operations in connection with the identified thread $22(t_B)$ by receiving commands therefor (step 136).